

BEST AVAILABLE COPY

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-022369

(43)Date of publication of application : 21.01.1997

(51)Int.Cl.

G06F 11/28

G06F 9/46

G06F 9/46

(21)Application number : 07-172020

(71)Applicant : FUJITSU LTD

(22)Date of filing : 07.07.1995

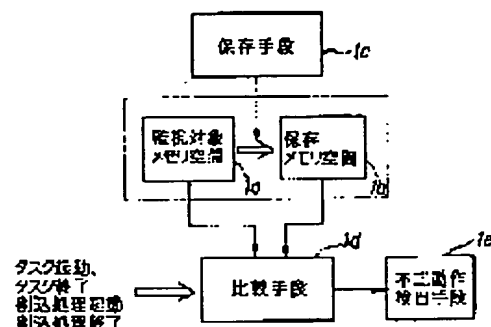
(72)Inventor : MASUDA HIDEJI

(54) ILLICIT OPERATION DETECTION METHOD IN KERNEL OF MULTI-TASKING SYSTEM

(57)Abstract:

PROBLEM TO BE SOLVED: To detect an illicit rewrite operation to a memory in terms of software and to search a task and a processing which performed the illicit rewrite operation.

SOLUTION: In this illicit operation detection method in the kernel of a multi-tasking system for executing a control transfer processing to a user interruption processing and a task dispatching processing for actuating a user task, a preservation means 1c preserves the contents of a monitoring object memory space 1a in a preservation memory space 1b beforehand, a comparison means 1d checks whether or not the memory contents of the monitoring object memory space 1a and the memory contents preserved in the memory 1b match with each other, as to the start of the user task, the processing end of the user task, the start of the user interruption processing and the end of user interruption processing, and an illicit operation detection means 1e discriminates a memory rewrite illicit operation by the user task or the user interruption processing based on the change of the memory contents.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2000 Japanese Patent Office

THIS PAGE BLANK (USPTO)

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平9-22369

(43) 公開日 平成9年(1997)1月21日

(51) Int.Cl. ⁸	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 11/28		7313-5B	G 0 6 F 11/28	A
9/46	3 3 0		9/46	3 3 0 C
	3 4 0			3 4 0 A

審査請求 未請求 請求項の数12 O L (全 17 頁)

(21) 出願番号 特願平7-172020

(22) 出願日 平成7年(1995)7月7日

(71) 出願人 000005223

富士通株式会社

神奈川県川崎市中原区上小田中4丁目1番
1号

(72) 発明者 増田 秀二

神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(74) 代理人 弁理士 斉藤 千幹

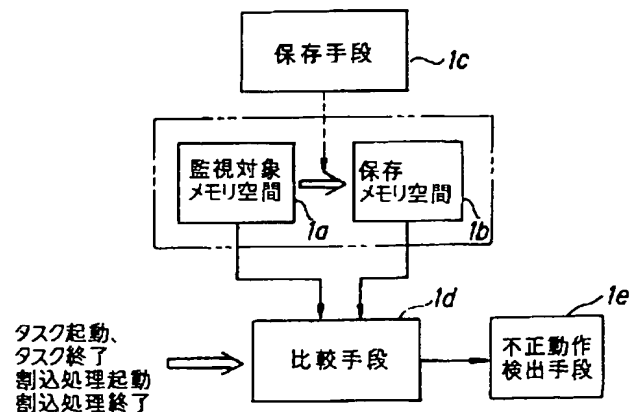
(54) 【発明の名称】 マルチタスキング方式のカーネルにおける不正動作検出方法

(57) 【要約】

【目的】 ソフト的にメモリへの不正な書替え動作を検出し、該不正書替え動作をしたタスク、処理を突き止める。

【構成】 利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、保存手段1cは予め監視対象メモリ空間1aの内容を保存メモリ空間1bに保存し、比較手段1dは、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、監視対象メモリ空間1aのメモリ内容とメモリ1bに保存してあるメモリ内容とが一致しているか調べ、不正動作検出手段1eはメモリ内容の変化に基づいて利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する。

本発明の第1の原理説明図



【特許請求の範囲】

【請求項1】 利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

予め設定されている監視対象メモリ空間の内容を保存する第1ステップ、

利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、監視対象メモリ空間のメモリ内容と保存してあるメモリ内容とが一致しているか調べる第2ステップ、メモリ内容の変化に基づいて所定の利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する第3ステップを有することを特徴とする不正動作検出方法。

【請求項2】 前記第1ステップは、監視対象メモリ空間全体の内容を加算し、あるいはその他の演算処理を施して該演算結果を保存し、

第2ステップは、監視対象メモリ空間全体の内容を加算し、あるいはその他の演算処理を施し、該演算結果と保存してある演算結果を比較することにより監視対象メモリ空間のメモリ内容の変化を調べることとする請求項1記載の不正動作検出方法。

【請求項3】 前記方法は更に、デバッガを設け、該デバッガを介して監視対象メモリ空間を設定するステップを有し、

前記第3ステップは、

メモリ内容の変化が検出された時、デバッガを起動するステップ、

デバッガによりメモリ内容を書き替えた利用者タスクの識別子あるいは利用者割込み処理の識別子を出力するステップを有することを特徴とする請求項1記載の不正動作検出方法。

【請求項4】 利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

予め、監視対象である入出力装置のアドレスと、該入出力装置にアクセス可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格とをそれぞれ設定する第1ステップ、

利用者タスクあるいは利用者割込み処理からカーネルの有する入出力処理機能がコールされた時、該コール要求に含まれる入出力アドレスと前記設定されているアドレスを比較する第2ステップ、

一致している場合には、該コール要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格を比較する第3ステップ、

識別子とアクセス資格の不一致に基づいて、入出力装置に対する利用者タスクあるいは利用者割込み処理の不正

アクセスを識別する第4ステップを有することを特徴とする不正動作検出方法。

【請求項5】 前記方法は、更に、デバッガを設け、該デバッガを介して前記監視対象である入出力装置のアドレスとアクセス資格を設定するステップを有し、

前記第4ステップは、

コール要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格の不一致が検出された時、デバッガを起動するステップ、

デバッガによりアクセスした利用者タスクの識別子あるいは利用者割込み処理の識別子を出力するステップを有することを特徴とする請求項4記載の不正動作検出方法。

【請求項6】 利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

予め、監視対象オブジェクトの識別子と該オブジェクトの操作が可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格とをそれぞれ設定する第1ステップ、

利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子と前記設定されているオブジェクト識別子を比較する第2ステップ、

一致している場合には、該オブジェクト操作要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格を比較する第3ステップ、

利用者タスクあるいは利用者割込み処理の識別子とアクセス資格との不一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する第4ステップを有することを特徴とする不正動作検出方法。

【請求項7】 前記方法は、更に、デバッガを設け、該デバッガを介して前記監視対象オブジェクトの識別子とアクセス資格を設定するステップを備え、

前記第4ステップは、

オブジェクト操作要求を発行した利用者タスクあるいは利用者割込み処理の識別子とアクセス資格の不一致が検出された時、デバッガを起動するステップ、

デバッガによりアクセスした利用者タスクの識別子あるいは利用者割込み処理の識別子を出力するステップを有することを特徴とする請求項6記載の不正動作検出方法。

【請求項8】 利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

予め、監視対象オブジェクトの識別子及び該オブジェクトに対する操作内容をそれぞれ設定する第1ステップ、

3

利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子、操作内容が前記設定されているオブジェ識別子、操作内容とそれぞれ一致しているか比較する第 2 ステップ、

両者の一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正動作を識別する第 3 ステップを有することを特徴とする不正動作検出方法。

【請求項 9】 前記方法は、更に、デバッガを設け、該デバッガを介して前記監視対象のオブジェクト識別子とオブジェクトに対する操作内容を設定するステップを備え、

前記第 3 ステップは、

前記両者の一致が検出された時、デバッガを起動するステップ、

デバッガによりアクセスした利用者タスクの識別子あるいは利用者割込み処理の識別子を出力するステップを有することを特徴とする請求項 8 記載の不正動作検出方法。

【請求項 10】 利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

予め、監視対象オブジェクトの識別子及びタスクの異常終了とみなす異常終了コードあるいは異常終了とみなさない異常終了コードをそれぞれ設定する第 1 ステップ、利用者タスクあるいは利用者割込み処理からのシステムコールにより要求されたオブジェクト操作に対する異常終了時、該システムコールに含まれるオブジェクト識別子が前記設定されているオブジェクト識別子と一致するか、及び、異常終了コードが前記設定されている異常終了コードと一致しているか比較する第 2 ステップ、

オブジェクト識別子及び異常終了コードの両方の一致に基づいて、あるいはオブジェクト識別子の一致、異常コードの不一致に基づいて利用者タスクあるいは利用者割込み処理の異常を検出する第 3 ステップを有することを特徴とする不正動作検出方法。

【請求項 11】 利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

監視対象メモリにおける各空き領域の内容を保存する第 1 ステップ、

利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、各空き領域の内容と保存してある各空き領域の内容とが一致しているか調べる第 2 ステップ、

少なくとも 1 つの空き領域における内容変化に基づいて利用者タスク又は利用者割込み処理によるメモリ書替え

4

不正動作を識別する第 3 ステップを有することを特徴とする不正動作検出方法。

【請求項 12】 前記第 1 ステップは、監視対象メモリにおける空き領域毎に、空き領域の内容を加算し、あるいはその他の演算処理を施して、該演算結果を保存し、第 2 ステップは、監視対象メモリにおける各空き領域毎に、空き領域の内容を加算し、あるいはその他の演算処理を施し、該演算結果と保存してある対応する空き領域の演算結果とを比較することにより監視対象メモリにおける各空き領域の内容が一致しているか調べることを特徴とする請求項 11 記載の不正動作検出方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は不正動作検出方法に係わり、特に利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法に関する。近年の通信制御装置では組み込み型の O/S（オペレーティングシステム）を使用してソフトウェアで処理を行うのが常識となっている。このような通信装置では汎用 CPU を使用し、O/S はリアルタイム性や高速応答性が要求されることから、イベントドリブン型のマルチタスキング処理が可能なカーネルが主流である。組み込み型の O/S においてもソフトウェアデバッグのためにデバッガが用意されているのが普通であり、デバッガの機能が充実していればソフト開発の効率化が可能となり、十分な試験を実施できることから装置の信頼性も向上する。本発明は、マルチタスク方式で最も厄介なバグの一つである、処理に無関係なタスクによるメモリの不正な書き替え等の検出方式に関するものであり、ディスパッチ処理において不正な動作の検出を行うため、特別なハードウェア機構を必要としない点が特徴である。

【0002】

【従来の技術】 マルチタスキング処理においては、複数のタスクが同時に動いて処理を実行する。かかるマルチタスキングによるデバッグ中に、O/S により共通に管理されている資源、例えばメモリの内容が不正に書き替えられることがある。かかる場合には、メモリを書き替えたタスクを発見する必要があるが、多数のタスクが並行して動作しているためソフト的にその発見をするのが難しかった。すなわち、不正にメモリの書き替えが行われた場合、書き替えられたアドレスに対するアクセスをソフトのみで検出する手段はなかった。このため、従来は、CPU のバス状態を監視するハードウェアを設け、該ハードウェアにより特定アドレスのアクセスを検出し、NMI 割り込み等でソフトに通知する機構を設けて特定アドレスに対してアクセスが行われたことを検出している。

【0003】 O/S が管理するオブジェクト（セマフ

オ、メールボックス、イベントフラグ等)に対する不正操作についても同様であり、ソフト的にオブジェクトに不正操作したタスクを発見するのが難しかった。このため、オブジェクトを管理しているメモリに対して前述のハードウェアにより不正な書き込みを行っている利用者タスクや利用者割込み処理を突き止める方法でデバッグを行っている。尚、システムコール・トレース機能を有するO/Sの場合には、トレース情報の解析により不正な操作を行っている処理を見つけることが行われている。

【0004】

【発明が解決しようとする課題】しかし、上記従来の方法では以下の問題点がある。すなわち、

ハードウェアによるメモリアクセス検出法は、メモリ不正書替え動作を行ったタスクや処理を発見するための周辺回路が複雑で高価になる問題がある。

又、ハードウェアによるメモリアクセス検出法は、メモリプールの空き領域のようにダイナミックに変化する領域の不正書き込みに対処できない問題がある。これは、メモリプールの場合空き領域のアドレスが変化し、予め監視すべき領域(アドレス)を特定できないからである。

更に、ハードウェアによるメモリアクセス検出法では、オブジェクトに対する正当なアクセスも検出してしまう問題がある。すなわち、所定のオブジェクトに対する操作が許されている利用者タスクや利用者割込み処理からの正当なアクセスをも検出する。このため、アクセス資格との連携による不正アクセスのみの検出が要望されている。

不正なオブジェクト操作をトレース情報で検出する方法は、バグの発生状況によっては必要なトレース情報が失われて(トレース領域サイズ不足により)、原因が究明できない問題がある。

又、トレース情報が多いと不正な操作を行っている部分を見つけるのに手間がかかる問題がある。

【0005】以上から本発明の第1の目的は、メモリへの不正な書替え動作をソフト的に検出できる不正動作検出方法を提供することである。本発明の第2の目的は、メモリプールの空き領域のようにダイナミックに変化するメモリ領域の不正書替え動作を検出できる不正動作検出方法を提供することである。本発明の第3の目的は、入出力装置に対する不正のアクセス動作を検出できる不正動作検出方法を提供することである。本発明の第4の目的は、アクセス資格との連携によりオブジェクトに対する不正な操作を検出できる不正動作検出方法を提供することである。

【0006】

【課題を解決するための手段】前記第1の目的は本発明によれば、図1に示すように、予め設定されている監視対象メモリ空間1aの内容を別のメモリ空間1bに保存

する手段1cと、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、監視対象メモリ空間1aのメモリ内容とメモリ空間1bに保存してあるメモリ内容とが一致しているか調べる手段1dと、メモリ内容の変化に基づいて所定の利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する手段1eとにより達成される。・・・請求項1

【0007】前記第2の目的は、本発明によれば、監視対象メモリにおける各空き領域の内容を保存する手段と、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、各空き領域の内容と保存してある各空き領域の内容とが一致しているか調べる手段と、少なくとも1つの空き領域における内容変化に基づいて所定の利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する手段とにより達成される。・・・請求項11

【0008】前記第3の目的は、本発明によれば、図2に示すように、予め、監視対象である入出力装置のアドレスAIOと、該入出力装置にアクセス可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格AQRとをそれぞれ設定する手段2aと、利用者タスクあるいは利用者割込み処理からカーネルの有する入出力処理機能がコールされた時、該コール要求に含まれる入出力アドレスと前記設定されているアドレスとを比較する第1の比較手段2bと、一致している場合には、該コール要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格AQRとを比較する第2の比較手段2cと、識別子とアクセス資格の不一致に基づいて、入出力装置に対する利用者タスクあるいは利用者割込み処理の不正アクセスを識別する手段2dとにより達成される。・・・請求項4

【0009】前記第4の目的は、本発明によれば、図3に示すように、予め、監視対象オブジェクトの識別子IDOと該オブジェクトの操作が可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格AQRとをそれぞれ設定する手段3aと、利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子と前記設定されているオブジェ識別子IDOとを比較する第1の比較手段3bと、一致している場合には、該オブジェクト操作要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格AQRとを比較する第2の比較手段3cと、利用者タスクあるいは利用者割込み処理の識別子とアクセス資格との不一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する手段3dとにより達成される。・・・請求項6

【0010】又、前記第4の目的は、本発明によれば、

7

予め、監視対象オブジェクトの識別子及び該オブジェクトに対する操作内容をそれぞれ設定する手段と、利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子、操作内容が前記設定されているオブジェクト識別子、操作内容とそれぞれ一致しているか比較する手段と、両者の一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する手段とにより達成される。・・・請求項 8

【0011】更に、前記第4の目的は、本発明によれば、予め、監視対象オブジェクトの識別子及びタスクの異常終了とみなす異常終了コードあるいは異常終了とみなさない異常終了コードをそれぞれ設定する手段と、利用者タスクあるいは利用者割込み処理からのシステムコールにより要求されたオブジェクト操作に対する異常終了時、該システムコールに含まれるオブジェクト識別子が前記設定されているオブジェクト識別子と一致するか、及び、異常終了コードが前記設定されている異常終了コードと一致しているか比較する手段と、オブジェクト識別子及び異常終了コードの両方の一致に基づいて、あるいはオブジェクト識別子の一致、異常コードの不一致に基づいて利用者タスクあるいは利用者割込み処理の異常を検出する手段とにより達成される。・・・請求項 10

【0012】

【作用】利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、保存手段 1 c (図 1) は予め監視対象メモリ空間 1 a の内容を保存メモリ空間 1 b に保存し、比較手段 1 d は、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、監視対象メモリ空間 1 a のメモリ内容とメモリ 1 b に保存してあるメモリ内容とが一致しているか調べ、不正動作検出手段 1 e はメモリ内容の変化に基づいて所定の利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する。この場合、保存手段 1 c は、監視対象メモリ空間 1 a 全体の内容を加算し、演算結果を保存メモリ空間 1 b に保存し、比較手段 1 d は、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、監視対象メモリ空間 1 a の全体の内容を加算し、加算結果と保存してある加算結果とを比較することにより監視対象メモリ空間 1 a のメモリ内容の変化を検出する。以上のようにすれば、保存メモリ空間のメモリ容量を少なくできる。・・・請求項 1、2

【0013】監視メモリ空間が、メモリプールのように空き領域が変化する場合には、各空き領域の内容を保存し、利用者タスクの起動、利用者タスクの処理終了、利

8

用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、各空き領域の内容と保存してある各空き領域の内容とが一致しているか調べ、少なくとも 1 つの空き領域における内容変化に基づいて所定の利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する。この場合、空き領域毎に空き領域の内容を加算し、加算結果を保存し、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、各空き領域毎に、該空き領域の内容を加算し、加算結果と保存してある対応する空き領域の加算結果とを比較することによりメモリ内容の変化を検出する。・・・請求項 11、12

【0014】利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、設定手段 2 a (図 2) は、予め、監視対象である入出力装置のアドレス A I O と、該入出力装置にアクセス可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格 A Q R とをそれぞれ設定し、第 1 比較手段 2 b は、利用者タスクあるいは利用者割込み処理からカーネルの有する入出力処理機能がコールされた時、該コール要求に含まれる入出力アドレスと前記設定されているアドレスを比較し、第 2 比較手段 2 c は、アドレスが一致している場合には、該コール要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格 A Q R を比較し、不正動作検出部 2 d は、識別子とアクセス資格の不一致に基づいて、入出力装置に対する利用者タスクあるいは利用者割込み処理の不正アクセスを識別する。・・・請求項 4

【0015】利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、設定手段 3 a (図 3) は、予め、監視対象オブジェクトの識別子 I D O と該オブジェクトの操作が可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格 A Q R とをそれぞれ設定し、第 1 比較手段 3 b は、利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子と前記設定されているオブジェクト識別子 I D O を比較し、識別子が一致している場合には、第 2 比較手段 3 c は、該オブジェクト操作要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格 A Q R を比較し、不正動作検出手段 3 d は、利用者タスクあるいは利用者割込み処理の識別子とアクセス資格との不一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する。・・・請求項 6

【0016】利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行す

るマルチタスキング方式のカーネルにおける不正動作検出方法において、予め、監視対象オブジェクトの識別子及び該オブジェクトに対する操作内容をそれぞれ設定し、利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子、操作内容が前記設定されているオブジェ識別子、操作内容とそれぞれ一致しているか比較し、両者の一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する。このようにしても、オブジェクトに対する不正操作を識別できる。・・・請求項8

【0017】又、予め、監視対象オブジェクトの識別子及びタスクの異常終了とみなす異常終了コード（あるいは異常終了とみなさない異常終了コード）をそれぞれ設定し、利用者タスクあるいは利用者割込み処理からのシステムコールにより要求されたオブジェクト操作に対する異常終了時、該システムコールに含まれるオブジェクト識別子が前記設定されているオブジェクト識別子と一致するか、及び、異常終了コードが前記設定されている異常終了コードと一致しているか比較し、オブジェクト識別子及び異常終了コードの両方の一致に基づいて（あるいはオブジェクト識別子の一致、異常コードの不一致に基づいて）利用者タスクあるいは利用者割込み処理の異常を検出する。・・・請求項10

【0018】

【実施例】

(A) メモリ不正書き換え動作検出

(a) メモリ不正書き換え動作検出タイミングの説明
図4は本発明のメモリの不正書き換え動作検出タイミングの説明図である。11はカーネルであり、タスクディスパッチ処理部11a、利用者割り込み処理への制御転送処理部11bから構成される。12、13は利用者タスク、15は利用者割込み処理、16は監視対象メモリである。イベントドリブン型のタスクスケジューリングでは、システムコール発行や外部割り込みにより、利用者タスク12、13の切替えや利用者割り込み処理15の実行が行われる。図4の例では次の順序で利用者タスクや利用者割込み処理が実行される。

【0019】処理(a)において、利用者タスク12からシステムコールAが発行されて利用者タスク13の起動が要求されると、タスクディスパッチ処理部11aはタスクスケジューリングアルゴリズムに従って利用者タスク13を起動する。利用者タスク13の処理(b)が終了すると、利用者タスク13はシステムコールBを発行してタスク終了をタスクディスパッチ処理部11aに通知する。これにより、タスクディスパッチ処理部11aは利用者タスク12を起動し、利用者タスク12はシステムコールAの直後から処理(c)を実行する。利用者タスク12の処理(c)の実行中に割り込みが発生すると、制御転送部11bは利用者割り込み処理15を起動す

る。利用者割り込み処理15の処理(d)が終了すると、利用者割り込み処理15は割込み処理終了を制御転送処理部11bに通知する。これにより、制御転送処理部11bは割り込み元である利用者タスク12を起動し、利用者タスク12は割込み発生直後から処理(e)を実行する。

【0020】以上から明らかなように、処理切替え毎にカーネル11のタスクディスパッチ処理あるいは制御転送処理が介入することになる。そこで、カーネル11の処理の切替えタイミングをメモリ不正書き込みの検出タイミングとし、該検出タイミングで監視対象メモリ16の内容変化を調べる。処理(a)～(d)のどこかでメモリの書き替えが行われれば、監視対象メモリの内容が変化し、書き替えたタスクや処理を検出できる。図4の例では、検出タイミングの時点のメモリ内容と検出タイミングの時点のメモリ内容が不一致となり、利用者タスク13の処理(b)の中で書き替えが行われたことが分かる。

【0021】(b) メモリの不正書き換え動作検出の第1実施例

図5は、利用者タスクによるメモリの不正書き換え動作検出の第1実施例説明図である。図中、11aはカーネルにおけるディスパッチ処理部、12～14は利用者タスク、16は監視対象メモリ、17はメモリ監視範囲の内容を複製される作業領域、18はメモリ監視範囲を記憶する監視範囲記憶部、19はデバッグ、20はコンソール装置である。

【0022】カーネルにより利用者タスク12が起動すると、(1)利用者タスク12はデバッグ強制起動割込みを発生し、デバッグ19を起動する。デバッグ19は強制起動割り込みにより起動されると、コンソール装置20を利用したコマンド入力が可能となる。(2)かかる状態において、コンソール装置20よりコマンド受け付け部19aを介してメモリ監視範囲が入力されると、デバッグ19はメモリ監視設定コマンド19bを実行して入力されたメモリ監視範囲を記憶部18に設定し、(3)同時に、指定範囲のメモリ内容を監視対象メモリ16から作業領域17に複製する。そして、デバッグ終了コマンドにより、利用者タスク12に復帰する。(4)利用者タスク12よりタスク切替のシステムコールが発生すると、(5)ディスパッチ処理部11aは、監視対象メモリ16と作業領域17の内容を比較する。(6)比較の結果、メモリ内容が一致していれば、ディスパッチ処理部11aは利用者タスク13を起動してタスクを切り替える。これにより、利用者タスク13の処理が開始する。

【0023】(7)利用者タスク13よりタスク切替のシステムコールが発生すると、(8)タスクディスパッチ処理部11aは前記と同様に監視対象メモリ16と作業領域17の内容を比較する。(9)メモリ内容が一致していれば、利用者タスク14にタスクが切り替わる。(10)

しかし、不一致であれば、タスクディスパッチ処理部11aは、利用者タスク14への切替を行わず、デバッグ19を起動する。デバッグ19が起動されることにより、利用者はコンソール装置20を利用して不正なメモリ書き込みを行った利用者タスクと書き込みを行ったプログラムの範囲を突き止めることができる。すなわち、カーネルは、不正動作が検出された時点（メモリ内容不一致検出の時点）で、該不正動作を行ったタスク識別子と検出部位（プログラム範囲）をログ収集して保持し、デバッグ19はコンソール装置20からの不正タスク表示要求により不正タスク表示コマンド19cを実行し、カーネルからログ収集結果を受信してコンソール装置20のディスプレイ画面に表示する。以上では、利用者タスクによる不正書替え動作検出を説明したが、利用者割込み処理による不正書替え動作検出も同様に行われる。

【0024】(c) メモリの不正書替え動作検出の第2実施例

図6は、利用者タスクによるメモリの不正書替え動作検出の第2実施例説明図である。図中、11aはカーネルにおけるディスパッチ処理部、12～14は利用者タスク、16は監視対象メモリ、17'はメモリ監視範囲の全内容を加算した加算結果を記憶する加算結果格納領域、18はメモリ監視範囲を記憶する監視範囲記憶部、19はデバッグ、20はコンソール装置である。第1実施例の場合、メモリ内容の比較において監視範囲が広いと作業領域17の確保が難しくなる。そこで、メモリ内容の変化を検出するための別の手段として、監視範囲内のメモリ内容の加算を行い、加算結果の比較によるメモリ内容変化検出法を用いる。第2実施例では、書き替えられた内容によっては変化を検出できない場合があるが、非常に稀にしか発生せず、メモリ不正書き込みの検出の目的には十分実用になる。

【0025】カーネル11により利用者タスク12が起動すると、(1)利用者タスク12はデバッグ強制起動割込みを発生し、デバッグ19を起動する。デバッグ19は強制起動割り込みにより起動されると、コンソール装置20を利用したコマンド入力が可能となる。(2)かかる状態において、コンソール装置20よりコマンド受け付け部19aを介してメモリ監視範囲が入力されると、デバッグ19はメモリ監視設定コマンド19bを実行してメモリ監視範囲を記憶部18に設定し、(3)同時に、指定範囲のメモリ内容を加算し、加算結果を加算結果記憶領域17'に格納する。しかる後、デバッグ終了コマンドにより、利用者タスク12に復帰し、該利用者タスクが実行される。(4)利用者タスク12よりタスク切替のシステムコールが発生すると、(5)ディスパッチ処理部11aは、監視対象メモリ16におけるメモリ監視範囲の内容を加算し、(6)該加算結果と加算結果格納領域17'に格納されている加算結果を比較する。(7)比較の結果、加算結果が一致していれば、ディスパッチ処理

部11aは利用者タスク13を起動してタスクを切り替える。(8)次に利用者タスク13よりシステムコールの発行によりタスク切替え要求が発生すると、(9)タスクディスパッチ処理部11aは同様に監視対象メモリ16におけるメモリ監視範囲の内容を加算し、(10)該加算結果と加算結果格納領域17'に格納されている加算結果を比較する。(11)比較の結果、加算結果が一致していれば、利用者タスク14に切り替わる。(12)しかし、不一致であれば、タスクディスパッチ処理部11aは、利用者タスク14への切替を行わず、デバッグ19を起動する。デバッグ19が起動されることにより、利用者は第1実施例の場合と同様にコンソール装置20を利用して不正なメモリ書替えを行った利用者タスクと書替えを行ったプログラムの範囲を突き止めることができる。以上では、加算を施した場合について説明したが、減算、排他的論理和演算その他の演算を施すことができる。

【0026】(B) 入出力装置に対す不正アクセス検出図7は入出力装置に対する不正アクセス検出の実施例である。図中、11はカーネルであり、ディスパッチ処理部11a、入出力処理部11cを有している。12～14は利用者タスク、19はデバッグ、20はコンソール装置、21は記憶部であり、コンソール装置よりデバッグを介して設定された監視対象の入出力装置のアドレスAIOと、該入出力装置にアクセス可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格AQRとを記憶する。カーネルの入出力処理機能が利用者タスクからCALLされると、同時に入出力装置のアドレスが通知されるので、このアドレスと設定されている監視対象の入力装置のアドレスAIOの比較する。一致すれば、アクセス資格AQRとカーネル11が認識している現在動作中の利用者タスク、利用者割込み処理等の識別子を比較する。一致しなければ、アクセス資格のないタスクあるいは処理から監視対象の入出力装置にアクセスが行われたことを検出できる。

【0027】以下、入出力装置に対す不正アクセス検出について詳細に説明する。カーネル11により利用者タスク12が起動すると、(1)利用者タスク12はデバッグ強制起動割込みを発生し、デバッグ19を起動する。デバッグ19は強制起動割り込みにより起動されると、コンソール装置20を利用したコマンド入力が可能となる。(2)かかる状態において、コンソール装置20よりコマンド受け付け部19aを介して監視対象である入出力装置のアドレスAIO(=100)、アクセス資格AQR(=利用者タスク14の識別子)が入力されると、デバッグ19は入出力監視設定コマンド19dを実行してこれらAIO、AQRを記憶部21に設定する。(3)そして、デバッグ終了コマンドにより、利用者タスク12に復帰する。(4)利用者タスク12よりシステムコールにより利用者タスク13の起動が要求されると、(5)

ディスパッチ処理部11aは、タスクスケジューリングアルゴリズムに従って利用者タスク13を起動し、処理を開始する。(6)そして、利用者タスク13が入出力アドレス=100として入出力処理機能をCALLすると、(7)カーネル11の入出力処理部11cは、不正アクセス判定処理を行う。

【0028】図8は不正アクセス判定処理のフローである。入出力処理機能がコールされると(ステップ101)、入出力処理部11cはCALLと同時に通知される入出力装置のアドレスと記憶部21に設定されている監視対象のアドレスAIOの比較を行う(ステップ102)。アドレスが一致しなければ、処理を継続し(ステップ103)、一致していれば、アクセス資格AQRとカーネル11が認識している現在動作中の利用者タスクの識別子を比較する(ステップ104)。一致すれば、正当なアクセスである。正当なアクセスの場合には、入出力処理実行後、利用者タスク13に復帰するので、不正なアクセスが発生しない限り利用者タスク13の処理は通常通りに実行され続ける(ステップ103)。一方、一致しなければ、アクセス資格のないタスクあるいは処理から監視対象の入出力装置にアクセスが行われたことになり(不正アクセス)、利用者タスクの実行を中断してデバッガ19を起動する(ステップ105、106)。

【0029】図7の例では、記憶部21に設定されているアクセス対象アドレスAIO(=100)と利用者タスク13から依頼された入出力アドレス(=100)と一致している。そこで、カーネル11が管理している動作中の処理(利用者タスク13)とアクセス資格(=利用者タスク14)を比較する。(8)一致していないため、入出力処理部11cは不正なアクセスと判断して、利用者タスクの実行を中断しデバッガを起動する。デバッガが起動されることにより、利用者はコンソール装置20を利用して不正動作を行ったタスクや、入出力処理をCALLしたプログラムを突き止めることができる。すなわち、カーネル11は、不正アクセス動作が検出された時点で、該不正アクセスを行ったタスクの識別子と検出部位(プログラム範囲)をログ収集して保持し、デバッガ19はコンソール装置20からの不正タスク表示要求によりカーネルからログ収集結果を受信してコンソール装置20のディスプレイ画面に表示する。

【0030】(C)オブジェクトに対する不正操作検出(a)オブジェクトに対する不正操作検出の第1実施例図9はオブジェクトに対する不正操作検出の第1実施例の構成図である。図中、11はカーネルであり、ディスパッチ処理部11a、オブジェクト操作処理部11dを有している。12~14は利用者タスク、19はデバッガ、20はコンソール装置、21は記憶部であり、コンソール装置よりデバッガを介して設定された監視対象オブジェクトの識別子IDOと、該オブジェクト操作が可

能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格AQRとを記憶する。オブジェクト不正操作検出の概略は以下の通りである。例えば、利用者タスク13からオブジェクト操作処理機能が呼出されると、同時にオブジェクト識別子が通知されるので、オブジェクト操作処理部11dはこの識別子と設定されている監視対象オブジェクトの識別子IDOを比較する。一致すれば、設定されているアクセス資格AQRとカーネルが認識している現在動作中の利用者タスクの識別子あるいは利用者割込み処理の識別子とを比較する。一致しなければ、アクセス資格のないタスクあるいは処理が監視対象オブジェクトを不正に操作したことを検出できる。

【0031】オブジェクトとしてはセマフォ、メールボックス、イベントフラグ、メモリプール等があり、O/Sにより管理されている。このうちセマフォは、整数を値としてとる変数で、P及びV命令によってのみ値が変更される。1つのセマフォに対しては、一時的にはただ1つのタスクだけがP又はV命令を実行できる。Sをセマフォ値とするとき、あるタスクがP(S)を実行すると、Sの値が1減らされた後、そのSの値が

$S \geq 0$ なら、そのタスクは実行を続ける。

$S < 0$ なら、そのタスクは停止し、Sに対する待ち行列に入る。

タスクV(S)を実行すると、Sの値が1増加された後、そのSの値が

$S > 0$ なら、そのタスクは実行を続ける。

$S \leq 0$ なら、Sに対する待ち行列から1つタスクを取り出し、そのタスクの実行を再開させる。V(S)を実行したタスク自身も実行を継続する。

【0032】図10はRMS 68Kのセマフォを用いた排他制御の説明図である。尚、ATSENは、そのタスクがある危険部分の排他制御のためにセマフォを用いることを宣言するもの、WTSENはセマフォを通して、危険部分の使用を要求するものであり、SGSEMはその使用を開放するものである。又、WTSENはセマフォ値を-1し、SGSEMは+1する。まずタスクbがセマフォを通して危険部分の使用を要求する。この場合、セマフォ値が1なので、その要求は受け付けられ、使用が許可される。そのとき、セマフォ値は-1されて0となる。しかる後、タスクaが走行し、同じセマフォに対して使用要求したとき、セマフォ値が1でないので、使用中であり、使用が許可されず、タスクaはセマフォキューに入る。さらに、タスクcが走行し、同じセマフォを要求するが、同様にセマフォキューにタスクcも入る。この時のセマフォ値は2回-1され、-2になっている。その後、タスクbがSGSEMにより使用を開放し、セマフォキューにあるタスクaが、その使用を許可される。更に、タスクaが開放することにより、タスクcの使用が許可され、最後にセマフォ値は1に戻り、セマフォキューも空となる。又、メールボックスはタスク間の交信手段であり、

あるタスクによってメッセージが置かれ、他のタスクによってメッセージが取り出される構造で表現された待ち行列である。メッセージを送る側のタスクはメールボックスに対してメッセージを送り、又、メッセージを受信する側のタスクはメールボックス上でメッセージの到着を待つ。かかるメールボックスを用いて同期制御や排他制御が行われる。

【0033】以下、図9に従ってオブジェクトの不正操作検出方法を詳述する。カーネル11により利用者タスク12が起動すると、(1) 利用者タスク12はデバッグ強制起動割り込みを発生し、デバッグ19を起動する。デバッグ19は強制起動割り込みにより起動されると、コンソール装置20を利用したコマンド入力が可能となる。(2) かかる状態において、コンソール装置20よりコマンド受け付け部19aを介して監視対象オブジェクトの識別子IDO(=28)、アクセス資格AQR(=利用者タスク14の識別子)が入力されると、デバッグ19はオブジェクト監視設定コマンド19eを実行してこれらIDO、AQRを記憶部21に設定する。(3) しかる後、デバッグ終了コマンドにより、利用者タスク12に復帰する。(4) 利用者タスク12よりシステムコールにより利用者タスク13の起動が要求されると、(5) ディスパッチ処理部11aは、タスクスケジューリングアルゴリズムに従って利用者タスク13を起動し、利用者タスク13は処理を開始する。(6) そして、利用者タスク13がシステムコール(オブジェクト識別子=23)を発行してオブジェクト操作を要求すると、(7) カーネル11のオブジェクト操作処理部11dは、不正操作判定処理を行う。

【0034】図11は不正操作判定処理のフローである。オブジェクト操作のシステムコールが発行されると(ステップ201)、オブジェクト操作処理部11dはシステムコールに含まれるオブジェクト識別子と記憶部21に設定されているオブジェクト識別子IDOとの比較を行う(ステップ202)。オブジェクト識別子が一致してなければ、処理を継続し(ステップ203)、一致していれば、アクセス資格AQRとカーネル11が認識している現在動作中の利用者タスクの識別子と比較する(ステップ204)。一致すれば、正当なオブジェクト操作である。正当なオブジェクト操作の場合には、オブジェクト操作処理後、利用者タスク13に復帰するので、不正なオブジェクト操作が発生しない限り利用者タスク13の処理は通常通りに実行され続ける(ステップ203)。一方、一致しなければ、アクセス資格のないタスクあるいは処理が監視対象のオブジェクトを操作したことになり(不正操作)、利用者タスクの実行を中断してデバッグ19を起動する(ステップ205、206)。

【0035】図9の例では、記憶部21に設定されているオブジェクト識別子IDO(=28)と利用者タスク

13が発行したシステムコールに含まれるオブジェクト識別子(=28)が一致している。そこで、カーネル11が管理している動作中の処理(=利用者タスク13)とアクセス資格(=利用者タスク14)を比較する。

(8) 一致していないため、オブジェクト操作処理部11dは不正なオブジェクト操作と判断して、利用者タスクの実行を中断しデバッグを起動する。デバッグ19が起動されることにより、利用者はコンソール装置20を利用して不正操作を行ったタスクやオブジェクト操作のシステムコールを発行したプログラム部分を突き止めることができる。すなわち、カーネル11は、オブジェクトの不正操作が検出された時点で、該不正操作を行ったタスクの識別子と検出部位(プログラム範囲)をログ収集して保持し、デバッグ19はコンソール装置20から不正タスク表示要求が入力されるとカーネルからログ収集結果を受信してコンソール装置20のディスプレイ画面に表示する。

【0036】(b) 第2実施例

図12はオブジェクト不正操作の検出の第2実施例の構成図である。11はカーネル、11eはイベント通知処理部、12は利用者カーソル、19はデバッグ、20はコンソール装置、21はオブジェクト操作検出条件が設定される記憶部、23はイベントフラグ管理情報記憶部である。第2実施例では、イベントフラグの不正操作を例にして説明する。又、フラグID=10のフラグに対し、本来セットされるはずのないビット7がセットされる現象が発生しているものとする。

【0037】カーネル11により利用者タスク12が起動すると、(1) 利用者タスク12はデバッグ強制起動割り込みを発生し、デバッグ19を起動する。デバッグ19は強制起動割り込みにより起動されると、コンソール装置20を利用したコマンド入力が可能となる。(2) かかる状態において、コンソール装置20よりコマンド受け付け部19aを介して監視対象オブジェクトの識別子IDO(=10)、機能FNC(=セットフラグ)、データDTA(=セットビット7)が入力されると、デバッグ19はオブジェクト操作条件設定コマンド19fを実行してこれらIDO、FNC、DTAを記憶部21に設定する。(3) そして、デバッグ終了コマンドにより、利用者タスク12の実行を再開させる。(4) 利用者タスク12がイベントフラグ操作のシステムコール(フラグID=10、セットフラグ、セットビット7)を発行してイベントフラグ操作を要求する。(5) カーネルのイベント通知処理部11eは、セットフラグのシステムコールが発行されると、デバッグの設定条件をもとに不正操作検出処理を実行する。

【0038】図13はイベントフラグの不正操作検出の処理フローである。オブジェクト操作(セットフラグ)のシステムコールが発行されると(ステップ301)、イベント通知処理部11eは、記憶部21に設定されて

いるオブジェクト操作条件（イベントフラグ操作条件）を参照する（ステップ302）。ついで、システムコールに含まれるオブジェクト操作条件と設定されているオブジェクト操作検出条件との比較を行う（ステップ303）。すなわち、システムコールに含まれるオブジェクト識別子（フラグID=10）、機能コード（=セットフラグ）、データ（=セットビット7）と、記憶部21に設定されているオブジェクト識別子IDO（=10）、機能コードFNC（=セットフラグ）、データ（=セットビット7）とをそれぞれ比較する。

【0039】オブジェクト操作条件が異なれば、正当なオブジェクト操作であり、イベントフラグ操作処理後に利用者タスク13に復帰し、利用者タスク13の処理を継続する（ステップ304、305）。一方、オブジェクト操作条件が一致すれば、利用者タスク12が設定されているイベントフラグの不正操作をしたことになり

（不正操作）、利用者タスクの実行を中断してデバッグ19を起動する（ステップ306、307）。図12の例では、オブジェクト操作検出条件が一致するから、イベント通知処理部11eは利用者タスク12の実行を中断し、デバッグ19を起動することになる。デバッグが起動されることにより、利用者はコンソール装置20を利用して不正操作を行ったタスクやシステムコールを発行したプログラム範囲を突き止めることができる。

【0040】(c) オブジェクト不正操作検出の第3実施例

図14はオブジェクト操作のシステムコールが単純に異常終了したことによりタスクの異常検出を行う第3実施例の構成図である。図中、11はカーネル、11fはシステムコール処理部、12は利用者タスク、19はデバッグ、21はオブジェクト操作検出条件が設定される記憶部である。デバッグ19は既に説明したように強制起動により起動する。そして、(1) デバッグ19はオブジェクト操作条件設定コマンドに従ってオブジェクト操作検出条件を記憶部21に設定する。第3実施例は、オブジェクト操作のシステムコールが、単純に異常終了するとタスクに異常ありと判定するものである。このため、オブジェクト操作検出条件におけるオブジェクトの種類／オブジェクト識別子として「全種類」が、システムコール異常終了コードとして「全種類」がそれぞれ設定される。

【0041】デバッグによるオブジェクト操作検出条件設定が終了すれば、利用者タスク12の処理が開始する。そして、(2) 利用者タスク12がオブジェクト操作のシステムコールを発行して所定のオブジェクト操作を要求すると、(3) カーネル11のシステムコール処理部11fは要求されたオブジェクト操作を実行しようとする。しかし、要求されたオブジェクト操作が実行できない場合、例えば、セマフォを獲得したいけれどセマフォがない、あるいは、領域を獲得したいけれど領域がない

等により要求されたオブジェクト操作が実行できない場合には、システムコール異常終了とする。(4) かかる異常終了により、システムコール処理部11fは記憶部21に設定されているオブジェクト操作検出条件を参照する。

【0042】オブジェクト操作検出条件において、オブジェクトの種類／オブジェクト識別子は「全種類」、システムコール異常終了コードは「全種類」となっている。このため、システムコール処理部11fは、システムコールで要求されたオブジェクトの種類や異常終了コードに関係なく、直ちに利用者タスク12の実行を中断し、デバッグ19を起動する。以後、利用者はコンソール装置20を利用して異常終了したタスクを突き止めることができる。

【0043】(d) オブジェクト不正操作検出の第4、第5実施例

図15は、指定したオブジェクト操作が異常終了したことによりタスクの異常を検出する第4実施例の構成図である。第4実施例において、第3実施例（図14）と異なる点は、オブジェクト種別／オブジェクト識別子を指定すると共に、タスクの異常とみなす異常終了コードを指定している点である。予め、監視対象オブジェクトの種類／識別子（イベントフラグ／識別子=10）及びタスクの異常とみなす異常終了コード（コード=8）をそれぞれ記憶部21に設定しておく。利用者タスク12からのシステムコールにより要求されたオブジェクト操作に対する異常終了時、イベント通知処理部11eは、該システムコールに含まれるオブジェクトの種別／識別子が前記設定されているオブジェクトの種別／識別子と一致するか、及び、異常終了コードが前記設定されている異常終了コードと一致しているか比較する。オブジェクトの種類／識別子及び異常終了コードの両方が一致している場合には、イベント通知処理部11eは、システムコールを発行した利用者タスク12の異常とみなし、該タスクの実行を中断し、デバッグを起動する。以後、利用者はコンソール装置を利用して異常終了したタスクを突き止めることができる。

【0044】図16は、指定したオブジェクトが指定した異常終了コード以外の異常で終了したことによりタスクの異常を検出する第5実施例の構成図である。第5実施例において、第3実施例（図14）と異なる点は、オブジェクト種別／オブジェクト識別子を指定すると共に、タスクの異常とみなさない異常終了コードを指定している点である。予め、監視対象オブジェクトの種類／識別子（イベントフラグ／識別子=10）及びタスクの異常とみなさない異常終了コード（コード=7）をそれぞれ記憶部21に設定しておく。利用者タスク12からのシステムコールにより要求されたオブジェクト操作に対する異常終了時、イベント通知処理部11eは、該システムコールに含まれるオブジェクトの種類／識別子が

前記設定されているオブジェクトの種類／識別子と一致するか、及び、異常終了コードが前記設定されている異常終了コードと一致しないか比較する。オブジェクトの種類／識別子が一致し、異常終了コードが不一致の場合、イベント通知処理部11eは、システムコールを発行した利用者タスク12の異常とみなし、該タスクの実行を中断し、デバッグを起動する。以後、利用者はコンソール装置を利用して異常終了したタスクを突き止めることができる。

【0045】(D)メモリアールの空き領域に対する不正書き込み動作の検出

図17は、メモリアールの空き領域に対する不正書き込み動作検出方式の説明図であり、監視対象メモリはメモリアール内の空き領域で、内容変化検出方式は加算結果比較方式である。図中、31はメモリアールで、左側はメモリブロック返却前のメモリアール、右側はメモリブロック返却後のメモリアールである。メモリアール31の空き領域はメモリブロック返却によりダイナミックに変化する。従って、メモリブロック返却ごとにメモリアールの空き領域31a~31cの内容加算を行い、加算結果を空き領域の最終アドレス31a'~31c'に最終ワードとして格納する。又、ディスパッチ処理部によるディスパッチ処理では、各空き領域毎に内容の加算を行い、加算結果と該空き領域の最終アドレスに格納されている加算結果とを比較し、一致／不一致により空き領域への不正書き込みの有無を判断する。

【0046】図18はメモリアールの空き領域に対する不正書き込み動作検出処理のフロー図である。タスク切替のシステムコールあるいは利用者割込み等が発生したか、換言すれば、検出タイミングになったかチェックする(ステップ401)。検出タイミングになれば、すなわち、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、メモリアール31の各空き領域毎に内容の加算を行い、加算結果と対応する空き領域の最終アドレスに格納されている加算結果とを比較する(ステップ402)。少なくとも1つの空き領域において加算結果が異なる場合には、空き領域への不正書き込みがあったものとみなし、利用者タスクの実行を中断し、デバッグを起動する(ステップ403、404)。以後、利用者はコンソール装置を利用してメモリアールの空き領域に不正書き込みをしたタスクあるいは処理を突き止めることができる。

【0047】一方、ステップ402において、全空き領域の加算結果が一致(空き領域の内容が一致)していれば、あるいは、ステップ401において、「NO」であれば、メモリブロックの返却(リリースブロック)によりメモリアールの空き領域が変化したかチェックする(ステップ405)、変化してなければ初めに戻り、ステップ401以降の処理を繰り返す。しかし、空き領域

が変化すれば、新たな空き領域の内容を加算し(ステップ406)、該加算結果を空き領域の最終アドレスに格納し(ステップ407)、以後、初めに戻り、ステップ401以降の処理を繰り返す。尚、演算は加算に限らず、減算、排他的論理和演算その他の演算を採用することができる。又、以上では空き領域の加算結果を比較して空き領域への不正書き込み動作を検出したが、空き領域の内容を保存しておき、保存内容と空き領域の内容を比較することにより不正書き込み動作を検出することもできる。以上、本発明を実施例により説明したが、本発明は請求の範囲に記載した本発明の主旨に従い種々の変形が可能であり、本発明はこれらを排除するものではない。

【0048】

【発明の効果】以上本発明は、異常となる現象は分かっているが、その現象を引き起こすタスクあるいは処理が不明な場合に適用して有効である。又、マルチタスキング方式のO/Sでは、タスクが数10本同時に動作しているのが普通なので、異常現象だけ分かっている発生元タスクを特定するのは困難なことが多い。本発明によれば、異常現象をブレイク条件として設定することにより、該現象が発生した時点で利用者タスクを中断してデバッグを起動するため、デバッグのメモリ表示機能等により異常現象の発生元タスクやシステムの状態を調査することができる。

【0049】本発明によれば、ソフト的にメモリへの不正な書替え動作を検出し、該不正書替え動作をしたタスク、処理を突き止めることができる。この結果、デバッグを効率良く行うことができる。又、この場合、監視メモリ領域の内容変化、すなわち、監視メモリ領域への不正書替え動作を、該領域の内容を加算してその加算結果の比較により行うことにより、不正書替え動作検出に使用するメモリ容量を著しく少なくできる。本発明によれば、メモリアールの空き領域のようにダイナミックに変化するメモリ領域の不正書替え動作を検出し、該不正書替え動作をしたタスク、処理を突き止めることができる。この結果、デバッグを効率良く行うことができる。又、空き領域の内容変化を加算結果の比較により行うことにより、不正書替え動作検出に使用するメモリ容量を著しく少なくできる。

【0050】本発明によれば、アクセス資格との連携により、入出力装置に対する不正のアクセス動作やオブジェクトに対する不正なアクセス操作を検出し、該不正アクセス、不正操作をしたタスク、処理を突き止めることができる。この結果、デバッグを効率良く行うことができる。

【図面の簡単な説明】

【図1】本発明の第1の原理説明図である。

【図2】本発明の第2の原理説明図である。

【図3】本発明の第3の原理説明図である。

21

22

【図4】メモリ不正書替え検出タイミングの説明図である。

【図5】利用者タスクによるメモリの不正書替え動作検出の第1実施例である。

【図6】利用者タスクによるメモリの不正書替え動作検出の第2実施例である。

【図7】入出力装置に対する不正アクセス動作検出の実施例である。

【図8】I/O不正アクセスの判定処理フローである。

【図9】オブジェクトに対する不正操作検出の第1実施例である。

【図10】セマフォを用いた排他制御の説明図である。

【図11】オブジェクト不正操作の判定処理フローである。

【図12】オブジェクト不正操作検出の第2実施例である。

【図13】イベントフラグ不正操作検出の処理フローで

ある。

【図14】オブジェクト不正操作検出の第3実施例である。

【図15】オブジェクト不正操作検出の第4実施例である。

【図16】オブジェクト不正操作検出の第5実施例である。

【図17】メモリプールの空き領域に対する不正書替え検出の説明図である。

【図18】メモリプールの空き領域に対する不正書き込み動作検出の処理フローである。

【符号の説明】

1 a・・・監視対象メモリ空間

1 b・・・保存メモリ空間

1 c・・・保存手段

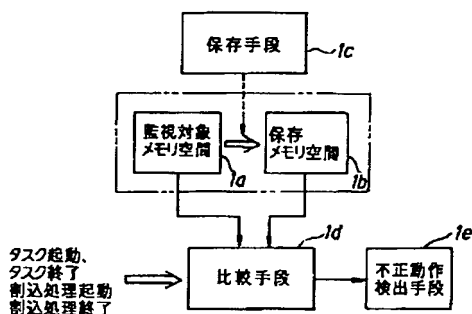
1 d・・・比較手段

1 e・・・不正動作検出手段

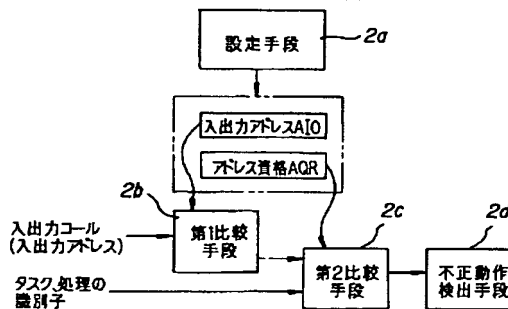
【図1】

【図2】

本発明の第1の原理説明図

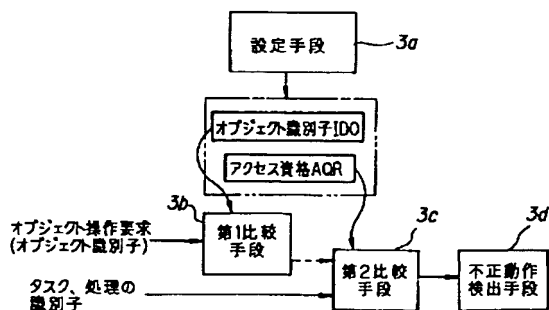


本発明の第2の原理説明図



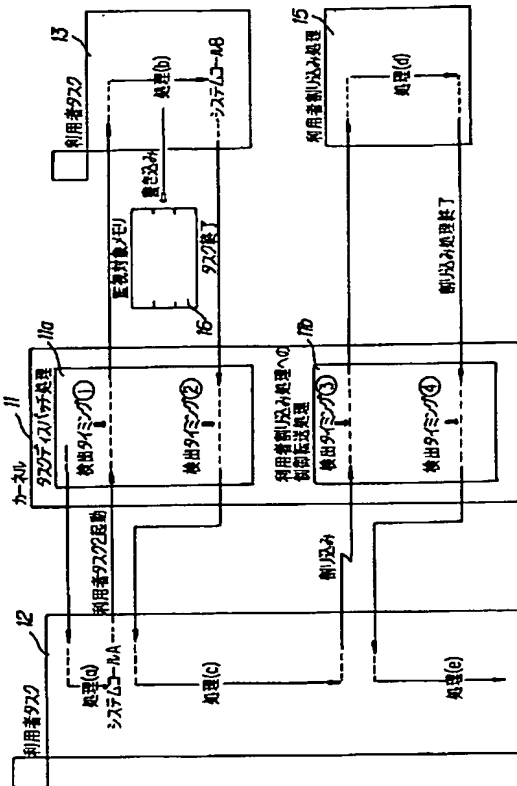
【図3】

本発明の第3の原理説明図



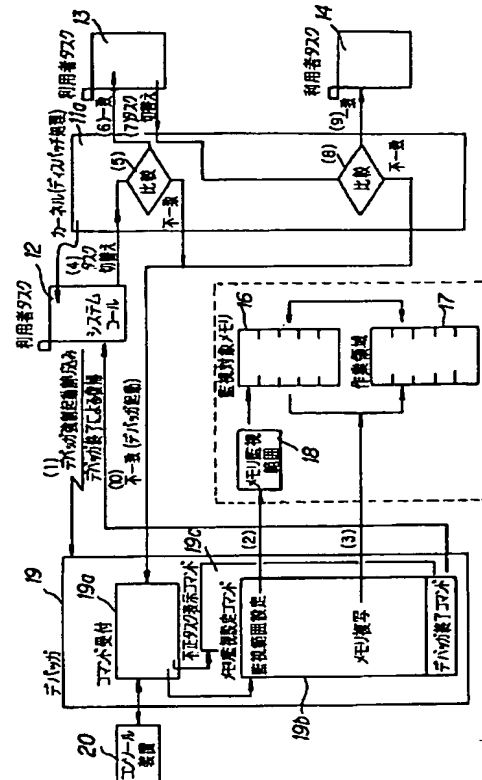
【图4】

メモリ書き替えの検出タイミング説明図



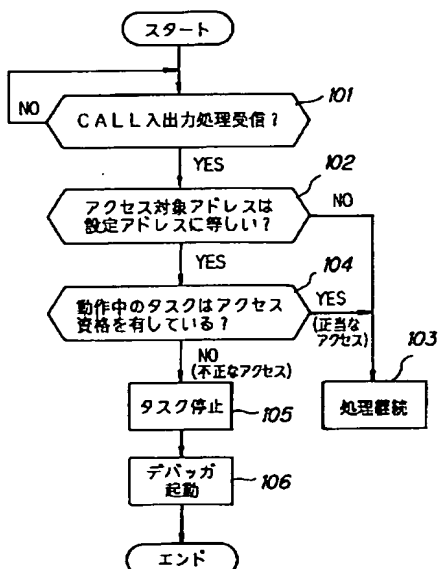
【 図 5 】

利用者タスクによるメモリ不正書き替え動作検出の第1実施例



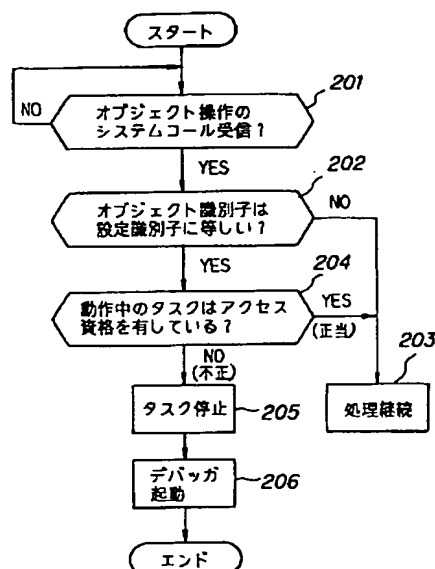
【图 8】

I/O 不正アクセス判定処理



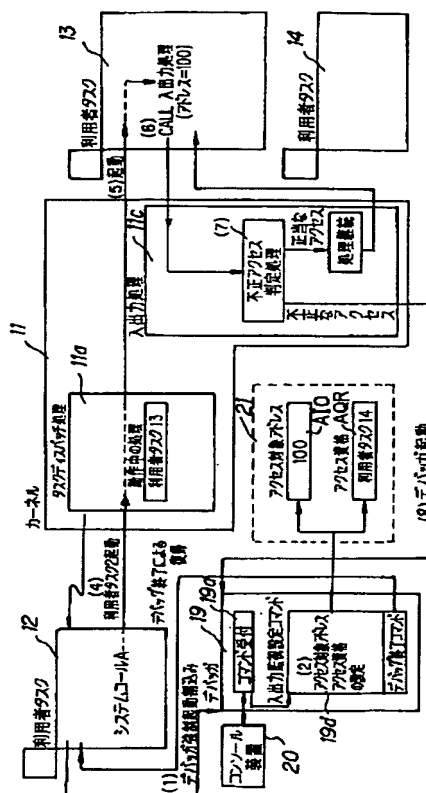
【 ㊦ 1 1 】

オブジェクト不正操作の判定処理

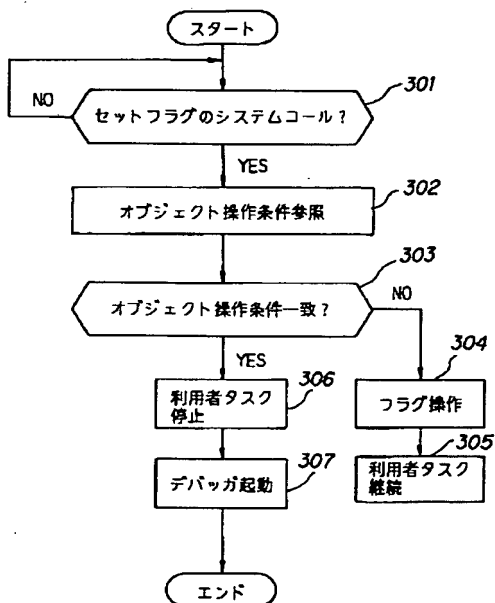


【图 7】

入出力装置に対する不正アクセス動作検出の実施例

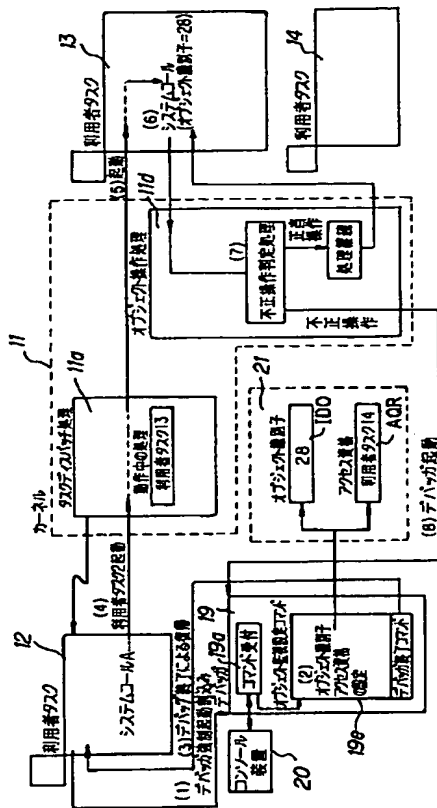


イベントフラグ不正操作検出処理



【図 9】

オブジェクトに対する不正操作検出の第1実施例

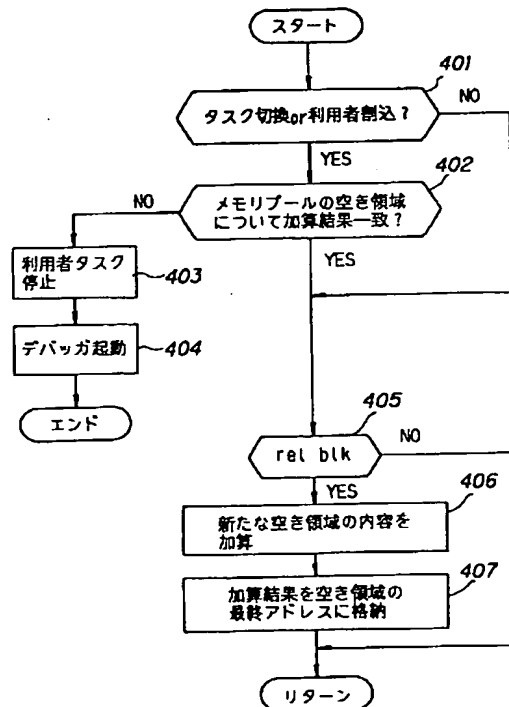


【図 10】

セマフォを用いた排他制御の説明

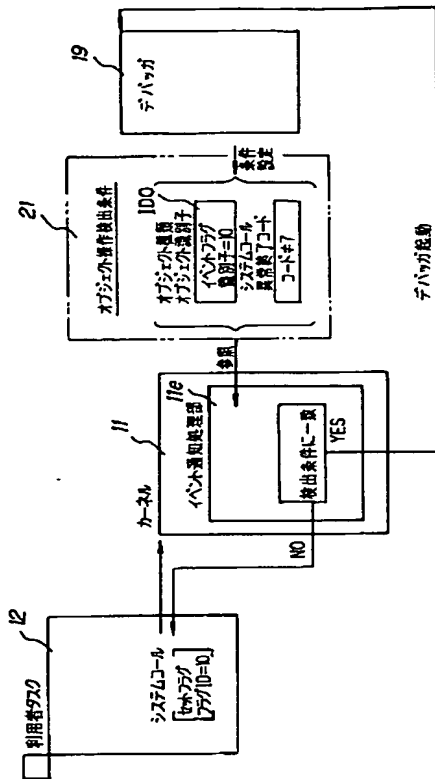
	タスクa	タスクb	タスクc	セマフォ値	セマフォキュー
初期				1	空
t ₁		ATSEM WTSEM 使用		0	↓
t ₂	ATSEM WTSEM			-1	タスクa
t ₃			ATSEM WTSEM	-2	タスクa タスクc
t ₄		SGSEM		-1	タスクc
t ₅	SGSEM		使用	0	↓
			SGSEM	1	空
				1	空

【図 18】

メモリアールの空き領域に対する不正書き込み
動作検出処理のフロー

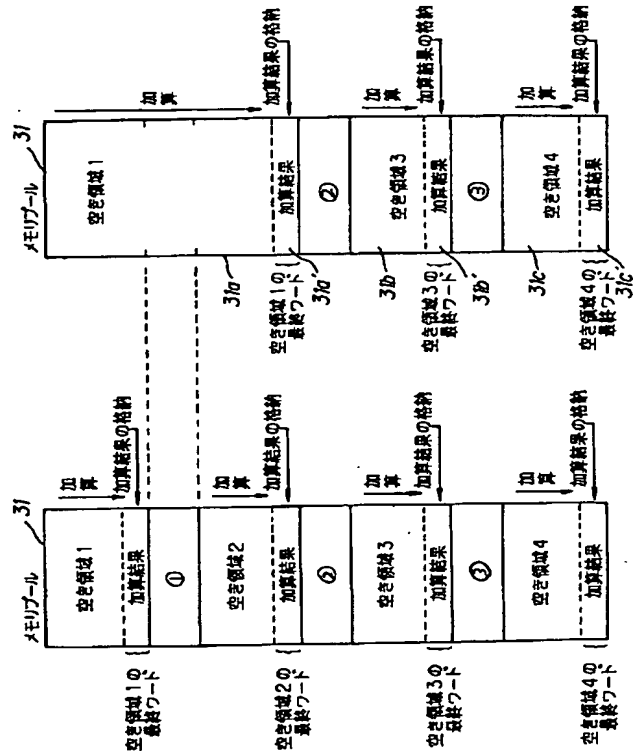
【図16】

オブジェクト不正操作検出の第5実施例



【図17】

メモリアールの空き領域に対する不正書き換え検出説明図



THIS PAGE BLANK (USPTO)